A Fast, High-Quality Inverse Halftoning Algorithm for Error Diffused Halftones

Thomas D. Kite, Niranjan Damera-Venkata, *Student Member, IEEE*, Brian L. Evans, *Senior Member, IEEE*, and Alan C. Bovik, *Fellow, IEEE*

***Abstract*—Halftones and other binary images are difficult to process with causing several degradation. Degradation is greatly reduced if the halftone is inverse halftoned (converted to grayscale) before scaling, sharpening, rotating, or other processing. For error diffused halftones, we present 1) a fast inverse halftoning algorithm and 2) a new multiscale gradient estimator. The inverse halftoning algorithm is based on anisotropic diffusion. It uses the new multiscale gradient estimator to vary the tradeoff between spatial resolution and grayscale resolution at each pixel to obtain a sharp image with a low perceived noise level. Because the algorithm requires fewer than 300 arithmetic operations per pixel and processes 7 × 7 neighborhoods of halftone pixels, it is well suited for implementation in VLSI and embedded software. We compare the implementation cost, peak signal-to-noise ratio, and visual quality with other inverse halftoning algorithms.**

***Index Terms*—Anisotropic diffusion, computational vision, image quality metrics, perceptually weighted noise measures.**

I. Introduction

HALFTONES and other binary images are difficult to process without causing severe degradation. Exceptions include cropping, rotation by multiples of 90°, and logical operations. Halftones are difficult to compress losslessly or lossily; grayscale images, on the other hand, can be compressed efficiently [1], [2]. Inverse halftoning permits a wide range of operations on halftones. Inverse halftoning recreates a grayscale image, with a typical wordlength of eight bits, from a halftone, with a wordlength of one bit. The problem is underdetermined—an essentially infinite number of possible grayscale images could have led to the given halftone, even if the halftoning method were known.

Screened halftones and error diffused halftones have greatly differing artifacts. As a consequence, inverse halftoning methods are generally tailored for either screened halftones or error diffused halftones. Some methods may be used for both types of halftones [3]–[5]. In this paper, we focus on error diffused halftones.

Published inverse halftoning methods for error diffused halftones include vector quantization [2], projection onto convex sets [6], MAP projection [7], nonlinear permutation filtering [3], Bayesian methods [8], and wavelets [5], [9]. Many methods show good results, but several are iterative, which require large amounts of computation and memory. Most also make heavy use of floating-point arithmetic. Among these algorithms, the wavelet algorithms [5], [9] arguably produce the best subjective results on error diffused images.

For error diffused halftones, we present a single-pass method with low computation and memory requirements that produces results comparable to those seen in the literature. The proposed method consists of multiscale directional gradient estimation followed by adaptive lowpass filtering. Most of the processing is accomplished with integer additions. The algorithm requires fewer than 300 arithmetic operations per pixel and only seven image rows are kept in memory at one time. It is well suited for implementation in VLSI and embedded software.

The proposed method is similar in spirit to the method proposed by Roetling [10]. While both methods attempt to produce an inverse halftone by using spatially varying linear filtering, there are several key differences. Our method is *single pass*, employs a highly sophisticated multiscale gradient estimator, and uses smoothing filters specifically tuned to the characteristics of error diffused halftones. Roetling's method estimates gradients from successive approximations to the grayscale image, which are formed by adaptively smoothing the halftone controlled by previous gradient estimates. The first estimate of the gradients is produced by computing pixel gradients from a lowpass filtered version of the halftone. Thus, the algorithm is iterative. Unlike [10], our multiscale gradient estimation is not only single pass but also tuned for error diffusion, and is much more robust to noise [11]. Also the bank of smoothing filters in [10] are not optimized for the characteristics error diffused halftones. Another key difference is that a plurality of smoothing filters in Roetling's method [10] are predetermined and stored, whereas the smoothing filters in the proposed method are data dependent and computed on the fly using a closed-form design formula that directly yields their filter coefficients. In summary, the proposed method is much more sophisticated and efficient (all of our filters were designed with implementation issues in mind) for inverse halftoning error diffused halftones than the algorithm in [10].

Manuscript received August 27, 1998; revised March 21, 2000. This work was supported by the U.S. National Science Foundation through an NSF CAREER Award under Grant MIP-9702707 and HP Laboratories. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jan P. Allebach.

T. D. Kite was with The University of Texas, Austin, TX 78712-1084 USA. He is now with Audio Precision, Beaverton, OR 97075-2209 USA (e-mail: tomk@ap.com).

N. Damera-Venkata is with the Laboratory for Image and Video Engineering, Department of Electrical and Computer Engineering, The University of Texas, Austin, TX 78712-1084 USA (e-mail: damera-v@ece.utexas.edu).

B. L. Evans and A. C. Bovik are with the Laboratory for Image and Video Engineering, Department of Electrical and Computer Engineering, The University of Texas, Austin, TX 78712-1084 USA (e-mail: bevans@ece.utexas.edu; bovik@ece.utexas.edu).

Publisher Item Identifier S 1057-7149(00)06916-5.

When we compare our algorithm against wavelet denoising, we choose [9] as the representative algorithm. Although Algorithm III in [5] is reported to have 0.2 dB higher peak signal-to-noise ratio (PSNR) for the *lena* image, we believe that the results of [9] are visually better. The algorithm in [9] uses large filters and floating-point arithmetic to compute the wavelet coefficients and stores the coefficients in nine floating-point images of the same size as the halftone. Our algorithm produces the same quality with an implementation cost that is several orders of magnitude lower. Our algorithm is ideally suited for low-resolution image binarization systems using error diffusion, such as low cost binary displays, and low resolution scanning applications [10].

Section II proposes the new inverse halftoning algorithm, which estimates local image gradients and uses these estimates to vary the cutoff frequency of a separable smoothing filter. Section III discusses and analyzes a fast implementation of the algorithm. Section IV compares the new algorithm with several existing methods in terms of visual quality, PSNR, and computational complexity. Section V concludes the paper. This paper is an expanded version of [12]. A C implementation of this algorithm is available at http://www.ece.utexas.edu/~bevans/projects/inverseHalftoning.html.

II. Proposed Algorithm

In inverse halftoning, a tradeoff between grayscale resolution and spatial resolution exists. Halftoning is essentially spatially-interactive wordlength reduction, usually from eight bits to one bit per pixel. Inverse halftoning can be viewed as spatially-interactive wordlength expansion. Averaging N binary samples produces a wordlength of $\log_2(N)$ bits; e.g., averaging 16 binary samples produces a four-bit wordlength. Because averaging blurs features within the support of the filter, a tradeoff exists between grayscale resolution (wordlength) and spatial resolution (detail). In inverse halftoning, the number of pixels in the halftone and the inverse halftone are equal. For an $M \times N$ image, 2^{MN} possible binary images and 256^{MN} possible 8-bit images exist. Since there is at most one unique grayscale image for a given deterministic inverse halftoning method, a maximum of 2^{MN} grayscale images from the much larger set of 256^{MN} possible images can be produced. Each of these images is therefore highly redundant.

A lowpass filter imposes a fixed relationship between the increase in grayscale resolution and decrease in spatial resolution. By *spatially varying* the tradeoff between increasing grayscale resolution and decreasing spatial resolution, we can obtain a large improvement in inverse halftone quality. In smooth regions, more pixels are included in the average, thereby increasing the wordlength. Near edges, fewer pixels are included in the average, thus preserving the edge. Our inverse halftoning algorithm obtains smooth regions (with many levels of gray) and sharp edges (with fewer levels of gray) by using spatially-varying linear filtering. The amount of smoothing performed by the filter at each pixel is controlled by a diffusion coefficient computed from the image gradients. The algorithm

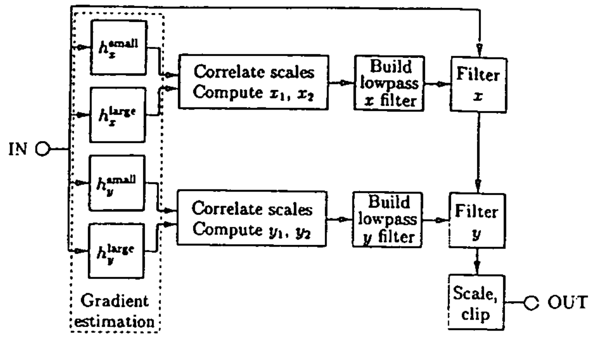

Fig. 1. Filtering in the inverse halftoning algorithm.

is a form of *anisotropic diffusion* [13], which was developed for robust multi-scale edge detection.

The basic idea in our proposed approach is to apply a separable linear filtering operation at each pixel in the halftone image. First, we design a highly customized family of smoothing filters, with frequency responses tailored to smoothing error diffused halftones. The family of filters is parameterized by a single parameter (control function). Our smoothing filter coefficients may be computed on the fly directly from the control function. Second, we design fixed gradient estimators for the x and y directions. These filters are designed to be bandpass in the direction of gradient estimation. Third, in order to increase the robustness of our directional gradient estimators to noise, we use two bandpass filters in each direction to detect small-scale and large scale edges, respectively. The gradient estimate is computed by combining the estimates of the two gradient estimation filters to maximize noise rejection. Fourth, we relate the directional multiscale gradient estimator outputs to the control function by a simple affine relationship. Last, at each pixel a separable smoothing filter is applied depending on the control function computed from the gradient estimator outputs.

Fig. 1 shows the steps in applying the spatially varying linear filter. Stage 1 computes gradients at two scales in both the horizontal (x) and vertical (y) directions. Stage 2 correlates the gradient estimates to give maximum output when a large gradient appears in both scales, such as at a sharp edge. We refer to the correlated estimates as *control functions*. Stage 3 constructs an FIR filter according the control functions. In each direction, the amount of smoothing increases as the estimated image gradient decreases. Smoothing occurs parallel to horizontal and vertical edges but not across them, thereby preserving edges in one direction while increasing grayscale resolution in the other. Stage 4 applies the FIR filter.

Section II-A discusses the design of a family of parameterized customized smoothing filters. These filters are designed to have low implementation complexity, and a frequency response tailored for error diffused halftones. Each filter in the family is designed to be parameterized by a single parameter, that controls its frequency response. Section II-B discusses the design of the multiscale gradient estimation filters. The output of these filters is combined to produce a control function that is able to choose the best parameterized smoothing filter at each pixel.

The inverse halftone is the result of applying appropriately (according to the control function) chosen smoothing filters at each pixel.

A. Smoothing Filter Design

We propose the following general criteria for the smoothing filter:

- FIR with small fixed size;
- simple to generate;
- separable;
- cutoff frequency determined by a single parameter;
- frequency response tailored for halftones.

Through testing on a set of eight natural images, we found that a 7 × 7 filter provided enough smoothing for good results.

Limit cycles are artifacts often present in halftones. Limit cycles should be suppressed in the inverse halftone; otherwise, they will lead to undesirable texture. In Floyd-Steinberg error diffusion, artifacts are particularly likely to occur at $(f_N, f_N), (f_N, 0)$, and, to a lesser extent, $(0, f_N)$ [14], where (f_h, f_v) denotes horizontal and vertical spatial frequency, respectively. We suppress these tones by placing zeros in the smoothing filter at these frequencies. Halftones produced using Jarvis error diffusion are less likely to contain these tones [14]. Because the smoothing filter is separable, a zero in the one-dimensional (1-D) prototype becomes a two-dimensional (2-D) (line) zero in the 2-D composite filter. By placing a zero at f_N in the x filter, for instance, we obtain a line zero at $(f_N, -)$ in the composite filter.

To preserve the image mean (brightness), the gain of the filter must be unity at dc, which constrains the filter at dc and f_N. We use a symmetric filter for linear phase [15]. We constrain the maximum passband ripple to ensure that the inverse halftone is a faithful reproduction of the original image. A filter with an excessively peaked passband produces falsely sharpened images. We found empirically that restricting the ripple to ±0.07 (±0.59 dB) produced high quality images that were not falsely sharpened. The maximum stopband gain was specified as 0.05 (−26 dB), so that the total noise power in the filter output decreases monotonically as the cutoff frequency of the filter is lowered. If the maximum stopband gain is not specified, then it is possible to design a filter h_1 whose cutoff frequency is lower than that of filter h_2, yet whose output has a higher noise power for the same input. This produces poor inverse halftones, since the reduction of quantization noise is no longer inversely proportional to the local image gradient. Since the smoothing filters are designed to be separable and linear phase, the coefficients in each dimension have the form $[a, b, c, d, c, b, a]$. By imposing the constraints at dc and f_N and minimizing the required computation, the filter response in each dimension is

$$h(n) = \frac{1}{4(x_2+2)} \times [x_2 - x_1 + 2, x_2, x_1, 4, x_1, x_2, x_2 - x_1 + 2] \quad (1)$$

where x_1 and x_2 are two parameters that must be chosen so that $h(n)$ satisfies the passband and stopband specifications. This is the *one-dimensional prototype* class. We construct two filters from the class at each pixel of the input image, one for each of the x and y directions. In the following analysis, we refer exclusively to the x filter. The y filter is constructed in the same way.

TABLE I
PARAMETERS OF THE SMOOTHING FILTERS

f_c (specified)	f_c (achieved)	f_s (achieved)	x_1	x_2
0.05	0.066	0.428	3.351	2.192
0.10	0.104	0.627	2.948	0.871
0.15	0.148	0.668	2.705	0.437
0.20	0.205	0.686	2.592	0.247
0.25	0.252	0.699	2.508	0.128
0.30	0.299	0.701	2.462	0.0326
0.35	0.352	0.793	2.145	-0.199
0.40	0.400	0.906	1.707	-0.427
0.45	0.455	0.930	1.452	-0.554
0.50	0.502	0.938	1.309	-0.621

We design a family of lowpass filters that meet the specifications. We employ sequential quadratic programming [16] to minimize the maximum stopband gain with respect to parameters x_1 and x_2, subject to a constraint on passband ripple. This leads to filters that are near-optimal in achieving the lowest transition width for the given filter size, passband ripple, and stopband gain. We design ten filters, each with a different desired cutoff frequency f_c, as shown in Table I. For each f_c, we fix the passband ripple at ±0.05 and adjust the stopband edge f_s to the lowest value possible, subject to a maximum stopband gain of 0.03. We find filters with the shortest transition width that satisfy the passband and stopband constraints.

Since we require the filter to be determined by a single parameter, we seek a functional relationship between x_1 and x_2 given by Table I. The lowest-order polynomial that gives an adequate fit for x_2 vs. x_1 is

$$x_2 = 0.4631x_1^3 - 2.426x_1^2 + 4.660x_1 - 3.612. \quad (2)$$

The filters in the continuous set defined by (1) and (2) have cutoff frequencies that vary from $0.066f_N$ to $0.502f_N$, unity gain at dc, and a zero at f_N. The maximum passband ripple is ±6.2% (±0.52 dB), and the maximum stopband gain is 0.045 (−27 dB). Thus, the performance of the entire family is within the original specifications, despite the approximation of (2).

Fig. 2(a) shows the original *Lena* halftone. Fig. 2(b)–(d) show the halftone filtered with three filters from the family, with the same f_c in the x and y directions. The suppression of the components at $(f_N, 0)$, and (f_N, f_N) is visible above the hat (where the checkerboard pattern at (f_N, f_N) is prominent) and in the cheek (where vertical stripes at $(f_N, 0)$ are objectionable). Notice the increasing smoothness of the filtered image with decreasing f_c. The shoulder in Fig. 2(d) is quite smooth, whereas the feathers and eyes in Fig. 2(b) are clear and sharp. Therefore, the filter family provides a range of smoothness needed to produce good inverse halftones.

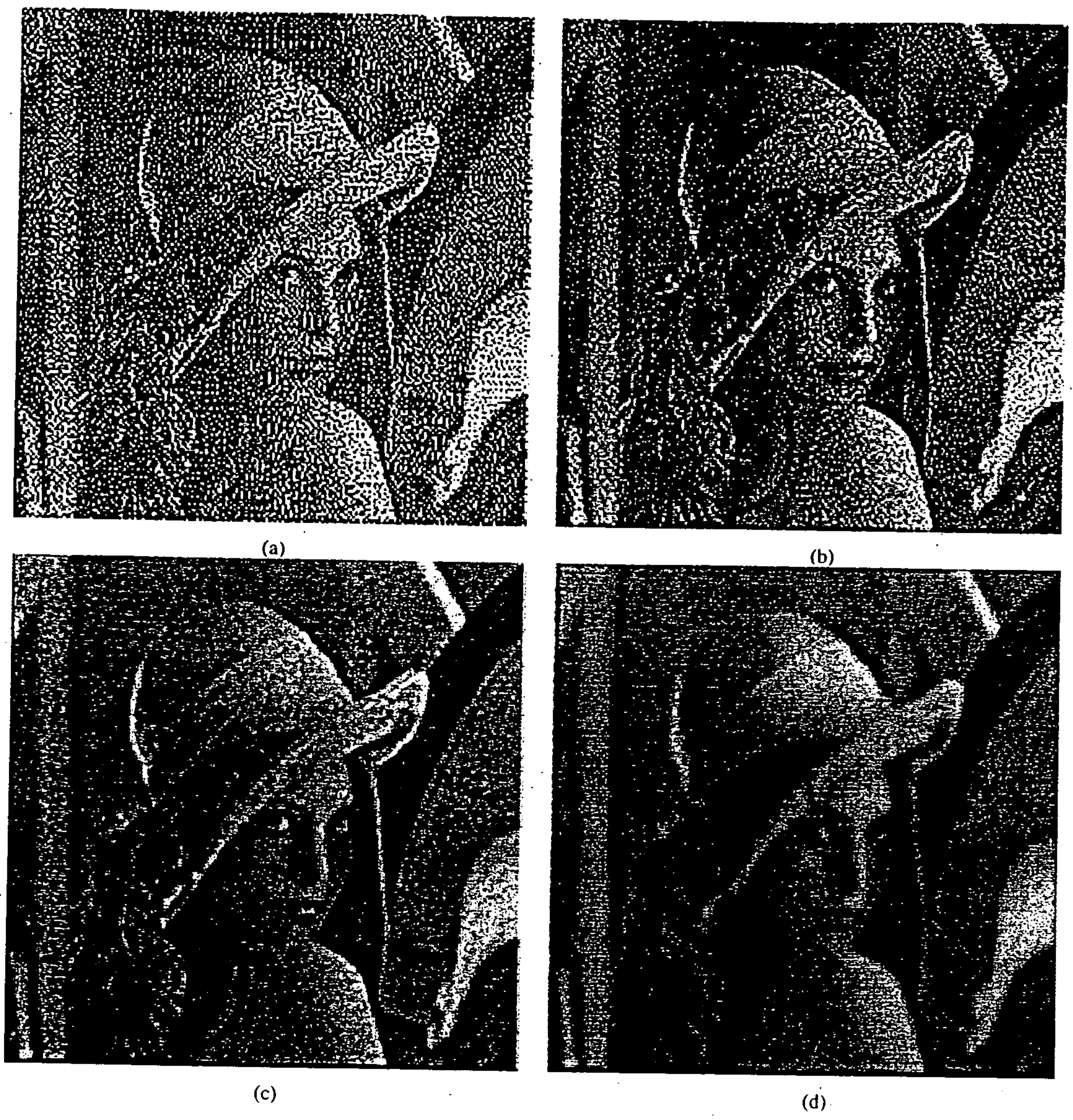

Fig. 2. Effect of the smoothing filter cutoff frequency on image smoothness. The Lena halftone is filtered with three different filters from the family using the given parameters. The filter parameter x_2 is computed from x_1 using (2). (a) Original Lena halftone. (b) $x_1 = 1.40 \Longleftrightarrow f_c = 0.46 f_N$. (c) $x_1 = 2.73 \Longleftrightarrow f_c = 0.15 f_N$. (d) $x_1 = 3.40 \Longleftrightarrow f_c = 0.065 f_N$.

B. *Gradient Estimator Design*

The Gaussian filter is the optimal presmoothing filter for gradient estimation in continuous signals in that it provides the best localization of gradients for a given range of scales [17]. In halftones, high-frequency quantization noise and strong idle tones introduce additional requirements on the presmoothing filter besides the conjoint minimization of spatial domain and frequency domain variances. We address the additional requirements in the design of our pre-smoothing filters. Although we make no claims about the optimality of these filters, we have found that they give better performance than Gaussians of the same size. The impulse responses of the resulting gradient estimators are very similar to those proposed as optimal by Canny [18].

To improve robustness to noise, we estimate gradients at two scales and correlate the results across scales. Large, sharp edges appear across scales, whereas noise does not [11]. For eight test images, gradient estimation at two scales gave the best

$$h_x^{\text{small}} = \frac{1}{1024}\begin{bmatrix} -19 & -32 & 0 & 32 & 19 \\ -55 & -92 & 0 & 92 & 55 \\ -72 & -120 & 0 & 120 & 72 \\ -55 & -92 & 0 & 92 & 55 \\ -19 & -32 & 0 & 32 & 19 \end{bmatrix}$$

$$h_x^{\text{large}} = \frac{1}{2048}\begin{bmatrix} -12 & -27 & -25 & 0 & 25 & 27 & 12 \\ -30 & -68 & -64 & 0 & 64 & 68 & 30 \\ -45 & -103 & -96 & 0 & 96 & 103 & 45 \\ -54 & -124 & -114 & 0 & 114 & 124 & 54 \\ -45 & -103 & -96 & 0 & 96 & 103 & 45 \\ -30 & -68 & -64 & 0 & 64 & 68 & 30 \\ -12 & -27 & -25 & 0 & 25 & 27 & 12 \end{bmatrix}$$

Fig. 3. Coefficients of the gradient estimation filters in the x direction. The superscripts "small" and "large" refer to the scale. The y filters are transposes of the x filters.

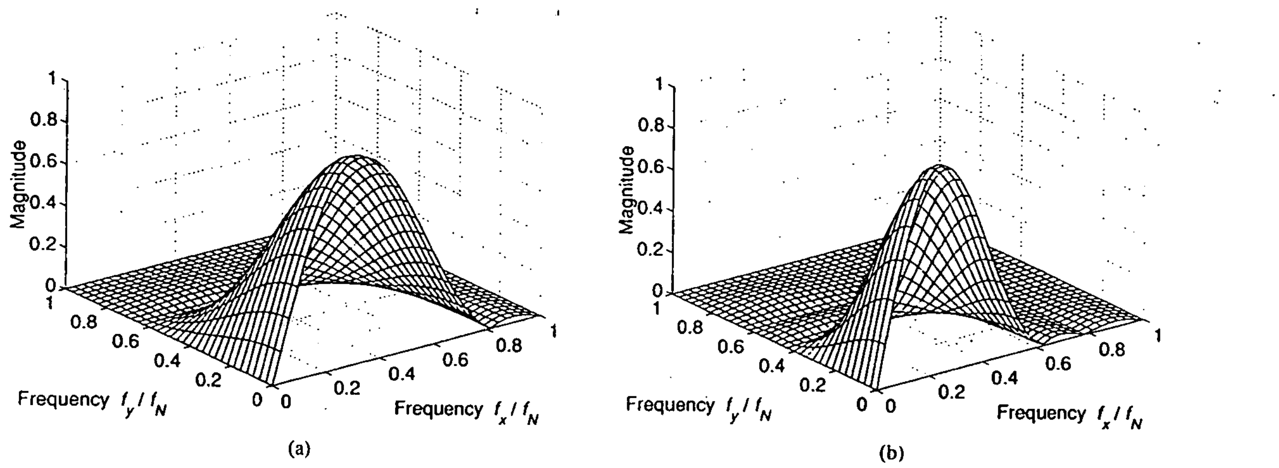

Fig. 4. Magnitude responses of the gradient estimation filters. The peak response and lowpass cutoff frequencies are approximately $0.32f_N$ and $0.090f_N$ for the small-scale estimator and $0.24f_N$ and $0.066f_N$ for the large-scale estimator. (a) h_x^{small} and (b) h_x^{large}.

performance. Using a third smaller scale increased noise in the inverse halftone. The specifications of the gradient estimation filters are

- line zeros at $(-,0), (f_N,-)$, and $(-,f_N)$;
- maximum stopband gain of 0.03;
- peak passband gain of 1;
- narrowest possible passband for a given filter size.

The filter passband is made as narrow as possible to best distinguish between the two scales. Each filter is separable. In the direction in which gradients are estimated, the filter is bandpass, with zeros at dc and the Nyquist frequency. In the direction perpendicular to the direction of gradient estimation, the filter is lowpass. The x filters are given in Fig. 3. The frequency responses of the two x filters are shown in Fig. 4. The near-linear rise of the response with frequency close to dc conforms to the $j\omega$ response of gradient estimators. We can see the line zeros at the band edges, and discern the equiripple behavior in the large-scale filter shown in Fig. 4(b).

At each pixel of the input image, we estimate gradients from the halftone using the filters h_x^{small}, h_y^{small}, h_x^{large}, and h_y^{large} to produce outputs e_x^{small}, e_y^{small}, e_x^{large}, and e_y^{large}, respectively. To correlate the gradients across scales, we compute the control functions according to the products

$$e_x^{\text{cf}} = \left|e_x^{\text{small}} \times e_x^{\text{large}} \times e_x^{\text{large}}\right|^{1/3},$$
$$e_y^{\text{cf}} = \left|e_y^{\text{small}} \times e_y^{\text{large}} \times e_y^{\text{large}}\right|^{1/3}. \tag{3}$$

We weight the large-scale gradients more heavily than the small-scale gradients to suppress small-scale noise. This produces slightly smoother, better quality inverse halftones than if equal weighting were used. Since each gradient estimator is linear, its output is proportional to its input. Each product in (3) is therefore proportional to the cube of the true image gradient. We find the cube root of the product so that the control function varies linearly with the gradient.

A perfect multiscale detector would produce identical estimates from both images. The output of a practical detector, however, is contaminated by noise in the halftone. This is demonstrated in Fig. 5, which shows gradients estimated from the original and halftoned versions of the *peppers* image. We

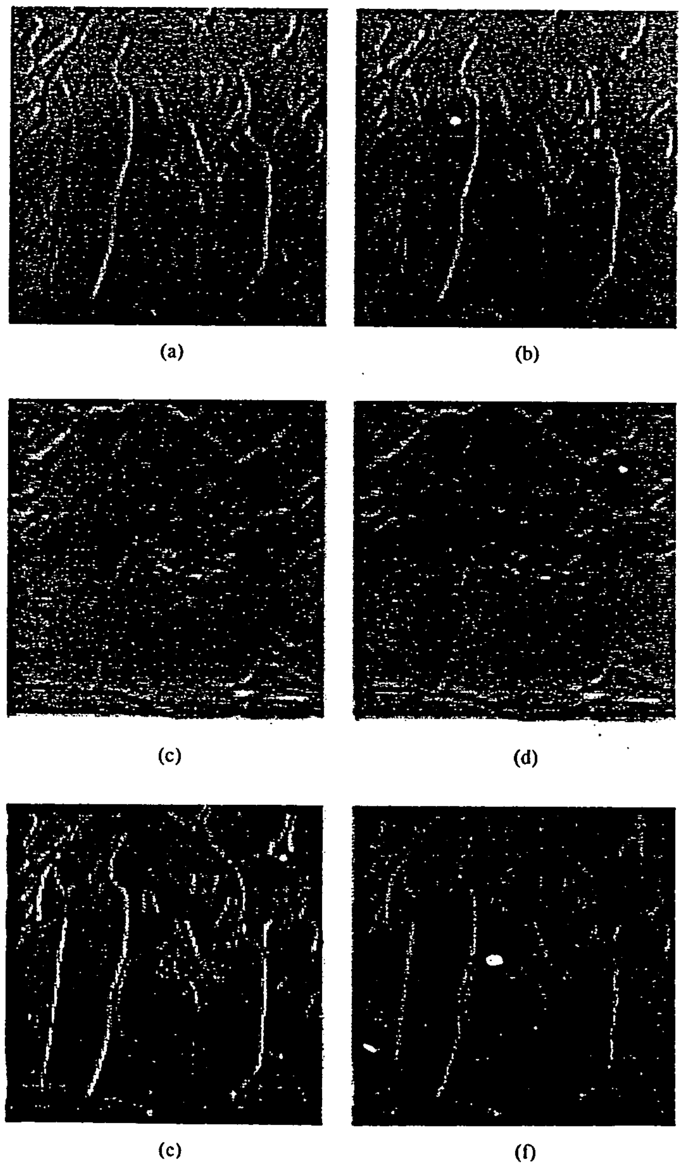

Fig. 5. Gradients estimated from *peppers* image. (a) e_x^{small} (original image); (b) e_x^{small} (halftone); (c) e_y^{large} (original image); (d) e_y^{large} (halftone); (e) e_x^{cf} (original image); and (f) e_x^{cf} (halftone).

use modified Floyd–Steinberg halftoning to give an unsharpened halftone [19].

Fig. 5(a) and (b) show the small-scale x direction gradients computed from the original image and halftone, respectively. Fig. 5(b) is noticeably noisier than Fig. 5(a) but also sharper. Fig. 5(c) and (d) show the large-scale y direction gradients computed from the original image and halftone, respectively. The noise is less noticeable in Fig. 5(d) than in Fig. 5(b) because the large-scale filter removes more of the quantization noise than the small-scale filter, but the image edges are not as sharp. Fig. 5(e) and (f) show the x direction control functions computed from the original image and halftone, respectively. By correlating across scales, we obtain most of the noise rejection of the large-scale gradient image, while retaining small-scale image edge information.

The x and y control functions, e_x^{cf} and e_y^{cf}, determine the cutoff frequencies of a separable smoothing filter, whose characteristics are described in Section II-A. We require a relation between e_x^{cf} and x_1. To reduce computation, we use the linear relation (the y relation is analogous)

$$x_1 = a + be_x^{\mathrm{cf}}. \qquad (4)$$

When the gradient magnitude is low, the image is smooth, and therefore the cutoff frequency of the lowpass filter should be low. This requires x_1 to be at the top of the allowable range: $x_1 \approx 3.4$ (see Table I). When the gradient magnitude is high, x_1 should be at the bottom of the allowable range: $x_1 \approx 1.4$. We start with $a = 3.4$, $b = -10$, and varied them while monitoring the visual quality of test images. The best results were achieved when $a = 3.33$ and $b = -5.7$.

III. Algorithm Implementation

To reduce computation, we compute x_2 from x_1 using Horner's form of (2)

$$x_2 = -3.612 + x_1(4.660 + x_1(-2.426 + 0.4631x_1)). \qquad (5)$$

We then construct the prototype filter according to (1), ignoring for the moment the factor of $1/(4(x_2 + 2))$. Each coefficient is a floating-point number in the approximate range $(-0.5, 4)$. We scale each coefficient by the factor 1024 (2^{10}), and convert it to an integer by discarding the fractional part. This results in at most a 13-bit signed integer, apart from the fixed central coefficient, which is 14-bit. The reason for this conversion is to permit application of the filter using integer arithmetic, which is quicker than floating-point arithmetic on most hardware.

The x and y prototype filters are applied separably to the 7×7 neighborhood centered on the current pixel. At the boundaries of the image, three pixels are replicated by mirroring to simplify the filtering. Applying the filters separably obviates the need to construct the equivalent 2-D filter. A 2-D filter would require 49 integer multiplications for its construction, and 48 integer additions for its application, per pixel. Applying the filters separably requires 42 integer additions in the x direction, followed by seven integer multiplications and six integer additions in the y direction, per pixel. Thus, 42 integer multiplications per pixel are saved.

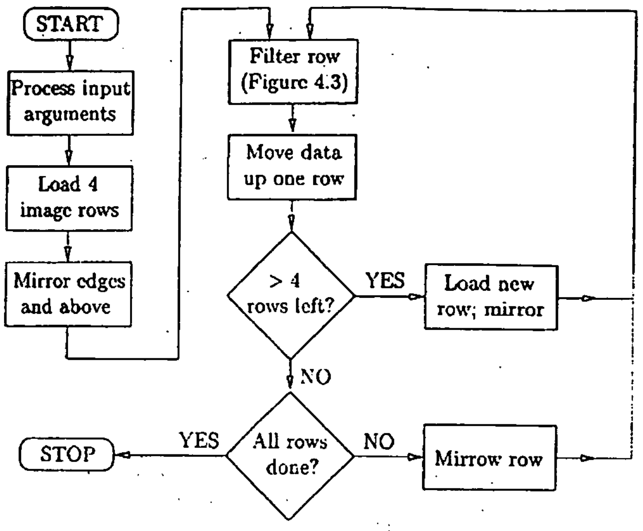

Fig. 6. Block diagram of the inverse halftoning algorithm. The filter applied at each pixel is determined by operation shown in Fig. 1. Because all operations are local, the algorithm is well-suited for implementation in VLSI or embedded software.

Each of the seven outputs of the x filter is at most a 16-bit signed integer; each is multiplied by one coefficient from the y filter, yielding at most a 29-bit signed integer product, apart from the central product, which may be 30-bit. The seven products are then summed, yielding at most a 32-bit signed result, which is a common integer wordlength for general purpose hardware. The coefficient quantization has no measurable effect on the final results.

The filtered output pixel is converted to a `float` and scaled. The scaling simultaneously accounts for the ignored factor $1/(4(x_2 + 2))$ in (1) (and the corresponding factor from the y filter), the scaling factor used in converting the filter coefficients to integers, and the requirement that the output pixels be in the range $(0, 255)$. Clipping enforces this range, before the pixel is rounded to the nearest integer and converted to an `unsigned char` (single byte).

Since the halftone is binary, we implement integer multiplication using integer addition. The number of integer additions depends on the image: 30 for all-black halftones, 128 for a mid-gray halftone, and 226 for all-white halftones. In computing the cube roots in (3) to derive the x and y control functions, we use bilinear approximation, followed by two iterations of Newton-Raphson approximation, which gives results accurate to better than 0.4%. For each cube root, we require a total of four additions, seven multiplications, and two divisions (all floating-point). We need three floating-point multiplications and additions for (5) in each direction. We need one floating-point division to normalize (1) in each direction. The arithmetic operations required per pixel are

- 30–226 integer additions;
- seven integer multiplications;
- 34 floating-point additions;
- 22 floating-point multiplications;
- six floating-point divisions;

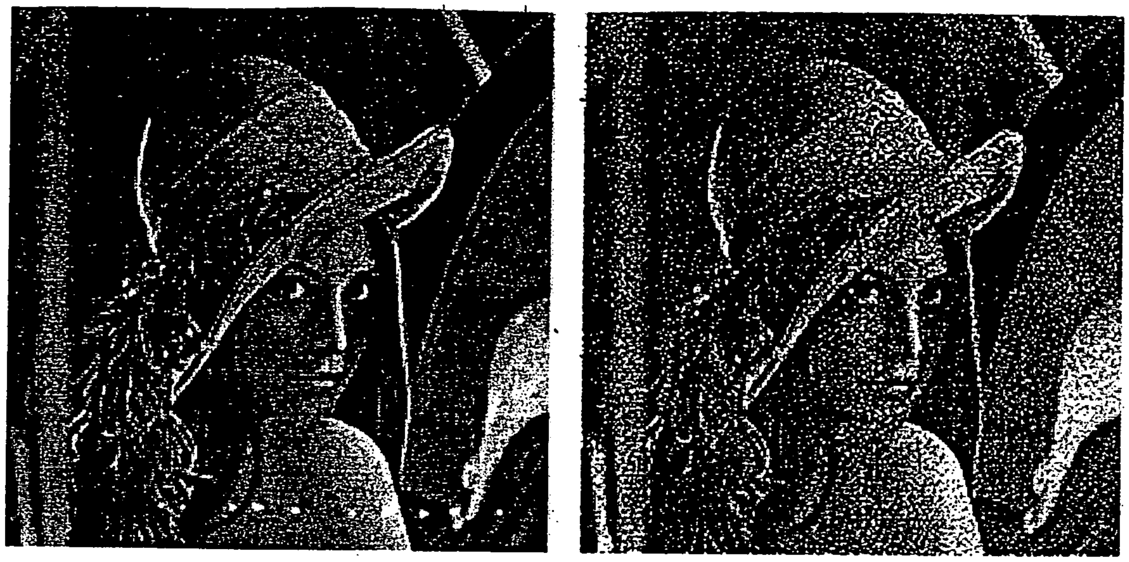
(a) (b)

Fig. 7. Original *Lena* image and its halftone. (a) Original image and (b) Floyd–Steinberg halftone.

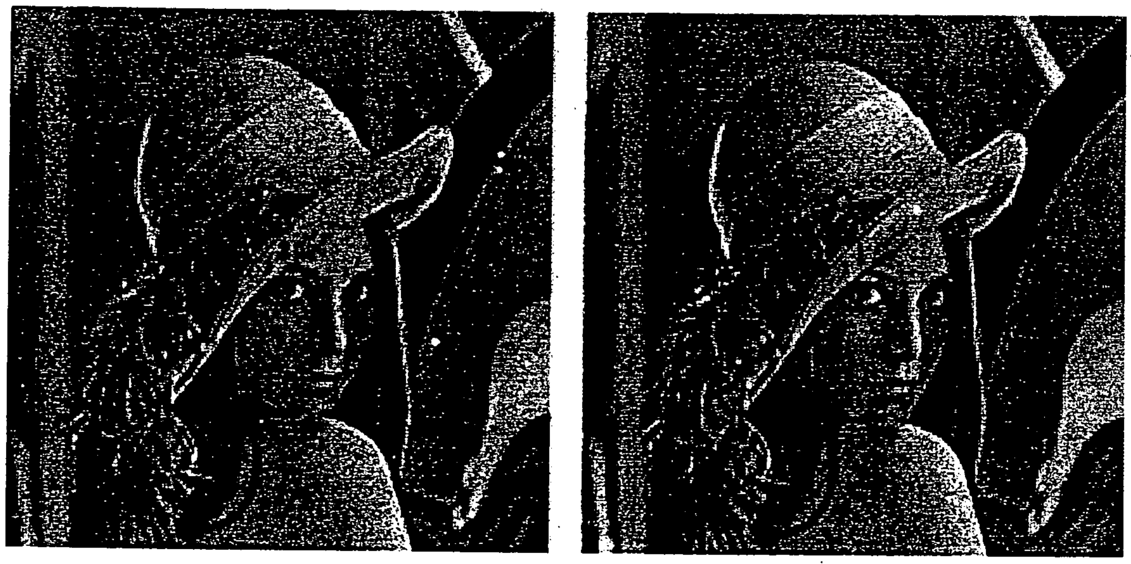
(a) (b)

Fig. 8. Inverse halftoned *Lena* images. (a) Proposed algorithm. PSNR 31.34 dB. (b) Wavelet algorithm, PSNR 31.47 dB.

The algorithm also requires 303 increment (++) operations. For a 512 × 512 image, algorithm executes in 2.9 s on a 167 MHz Sun UltraSparc-2 workstation. The Bayesian [8] and wavelet [9] algorithms require 18 s and 180 s, respectively [4]. All algorithms were implemented in C.

Execution proceeds in raster fashion, one row at a time. Seven image rows are required for the filters; they are kept in the *image storage area* of size $7(c + 6)$ bytes, where c is the number of image columns. There are six more columns in the storage area than in the image itself because of the mirroring extension of three pixels at the image boundaries. Each image pixel requires one byte of storage. For a 512 × 512 image, 3626 bytes of memory are allocated for image storage. A block diagram of the dataflow for an embedded (low memory) implementation of the proposed algorithm is shown in Fig. 6.

IV. Results

Fig. 7(a) shows the original *lena* image,[1] while Fig. 7(b) shows the Floyd–Steinberg halftone. Artifacts above the hat (containing tones close to (f_N, f_N)) and in the cheek (containing tones close to $(f_N, 0)$) are visible. Fig. 8(a) shows the result of inverse halftoning Fig. 7(b) using the proposed algorithm. The image shows a range of smooth and sharp areas; compare the the interior of the shoulder with that of its edge where it overlaps with the mirror. Artifacts are still

[1]Images referred to as "original" have been halftoned by the printing process used to render the figures on the paper. All of the images are therefore of low spatial resolution (512 × 512) and have been reproduced at as large of a dot size as possible to mitigate the effect of the printer. The images are best viewed by holding the page further from the eye until the halftone patterns due to the rendering vanish.

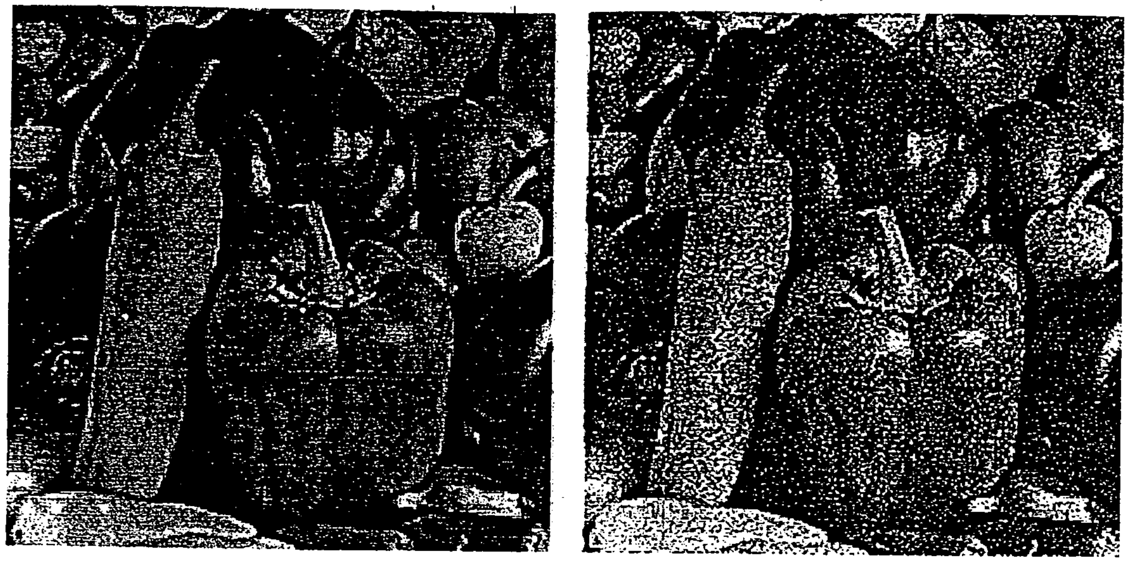

Fig. 9. Original *peppers* image and its halftone. (a) Original image and (b) Floyd–Steinberg halftone.

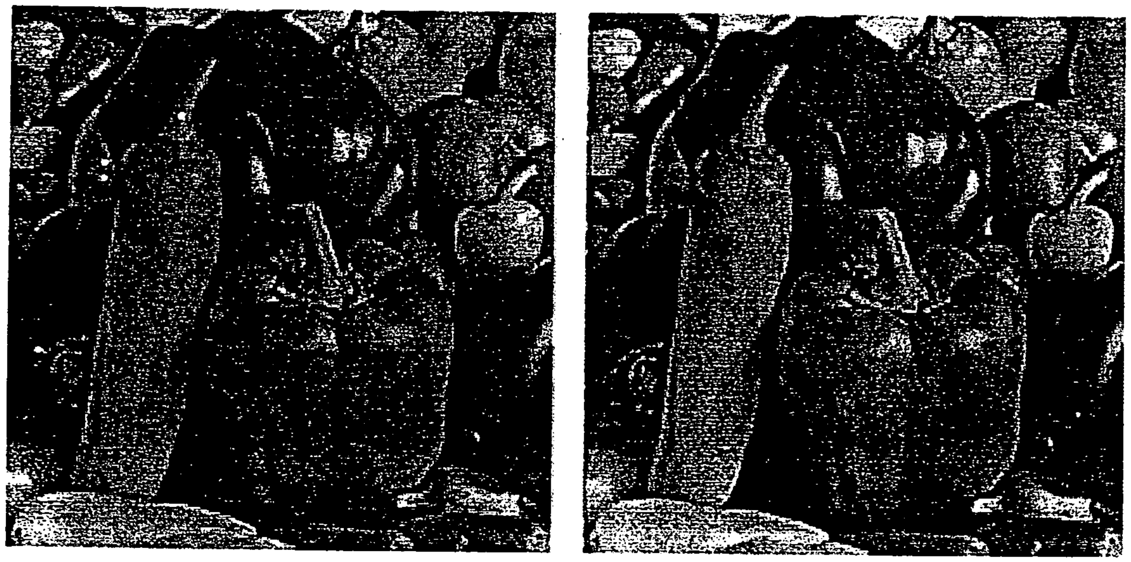

Fig. 10. Inverse halftoned *peppers* images. (a) Proposed algorithm, PSNR 31.43 dB. (b) Wavelet algorithm, PSNR 30.40 dB.

visible in the area above the hat, where the Floyd-Steinberg halftone is quasiperiodic. Fig. 8(b) shows the results of the wavelet-based algorithm. The wavelet image looks somewhat more natural, but the edges are not as sharp. The increased noise is particularly visible in the cheek and nose.

Fig. 9(a) shows the original *peppers* image, while Fig. 9(b) shows the Floyd-Steinberg halftone. Fig. 10(a) and (b) show the inverse halftones generated by the proposed method and the wavelet method, respectively. The image produced by the proposed method has sharper edges: the chile pepper at the left is more distinct, as is the stalk of the bell pepper.

Fig. 11(a) shows the original *Barbara* image, and Fig. 11(b) shows the Floyd-Steinberg halftone. Fig. 12(a) and (b) show the inverse halftones generated by the proposed method and the wavelet method, respectively. The *Barbara* image contains strong high frequencies that effectively cannot be recovered from the halftone, e.g., the stripes in the trousers. The proposed algorithm retains the sharp edges of the table leg and the books, and the skin on the face and arms is quite smooth. The edges in the wavelet image are not as sharp, and smooth areas are noisier.

Table II compares memory usage, computational complexity, and PSNR figures of merit for four inverse halftoning algorithms and the proposed algorithm. Table II shows that the proposed algorithm has very low memory requirements. The computational complexity of the proposed algorithm is also very low compared to methods that have comparable visual quality. The large improvement in PSNR for the *peppers* image is due in part to an error in the original image. This error was corrected for this work, and was reported to the authors of the wavelet-based method [9].

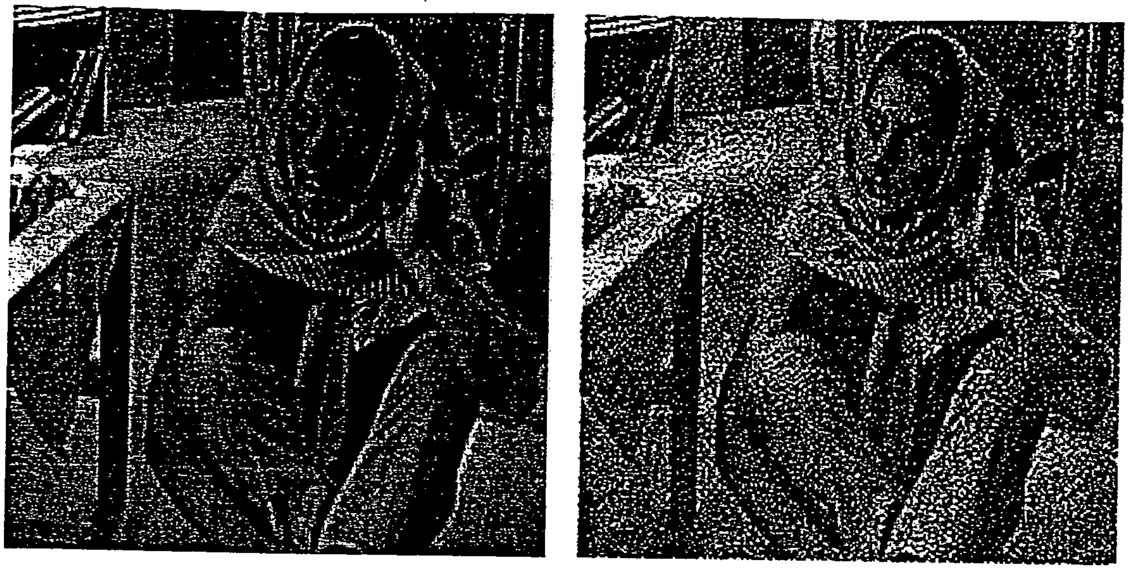
(a) (b)

Fig. 11. Original *Barbara* image and its halftone. (a) Original image and (b) Floyd–Steinberg halftone.

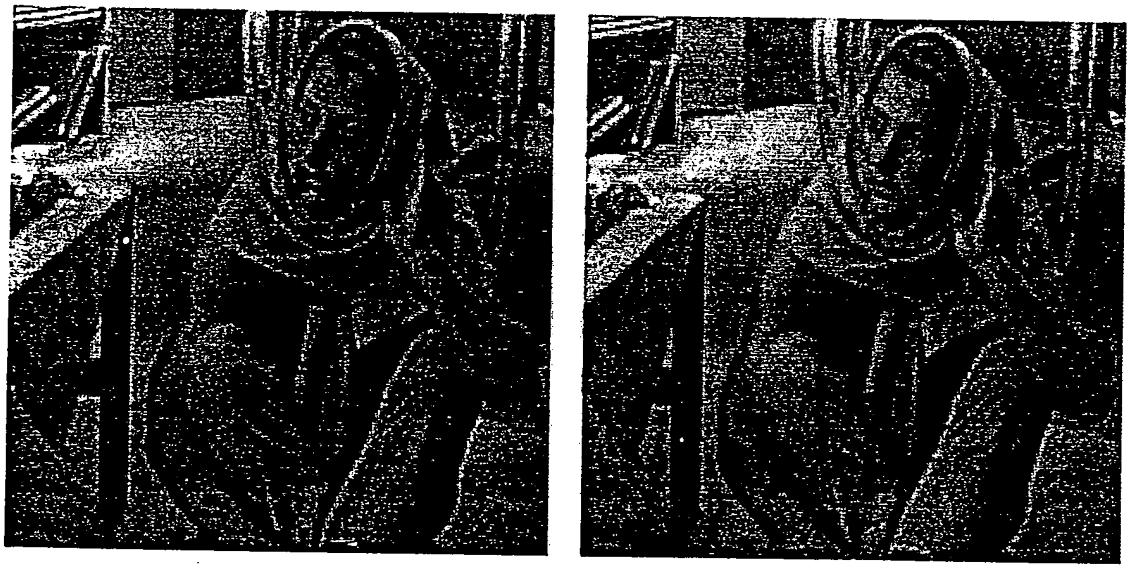
(a) (b)

Fig. 12. Inverse halftoned *Barbara* images. (a) Proposed algorithm, PSNR 24.61 dB. (b) Wavelet algorithm, PSNR 24.14 dB.

TABLE II
COMPARISON OF INVERSE HALFTONING METHODS FOR AN N × N HALFTONE

Algorithm & citation	Memory (bytes)	Estimated Complexity	PSNR (dB)	
			lena	*peppers*
Bayesian [8]	$8N^2$	High	—	—
POCS [6]	$8N^2$	High	30.4	—
Wavelet [9]	$36N^2$	Medium	31.5	30.4
Kernel est. [7]	$8N^2$	Medium	32.0	30.2
Proposed	$7N$	Low	31.3	31.4

V. CONCLUSION

We have presented 1) a fast inverse halftoning algorithm, and 2) a new multiscale gradient estimator for error diffused halftones. The algorithm produces high quality images from error diffused halftones at low computational cost. The control functions derived from the new multiscale gradient estimator determine the horizontal and vertical cutoff frequencies of a 7×7 smoothing filter applied to the halftone. The proposed algorithm not only achieves visual quality and PSNR results comparable with best algorithms in the literature, but does so at a fraction of the computation and/or memory cost.

REFERENCES

[1] D. Neuhoff and T. Pappas, "Perceptual coding of images for halftone display," *IEEE Trans. Image Processing*, vol. 3, pp. 1–13, Jan. 1994.

[2] M. Ting and E. Riskin, "Error-diffused image compression using a binary-to-grayscale decoder and predictive pruned tree-structured vector quantization," *IEEE Trans. Image Processing*, vol. 3, pp. 854–858, Nov. 1994.

[3] Y. Kim, G. Arce, and N. Grabowski, "Inverse halftoning using binary permutation filters," *IEEE Trans. Image Processing*, vol. 4, pp. 1296–1311, Sept. 1995.

[4] N. Damera-Venkata, T. D. Kite, M. Venkataraman, and B. L. Evans, "Fast blind inverse halftoning," in *Proc. IEEE Conf. Image Processing*, Oct. 1998, pp. 64–68.
[5] J. Luo, R. de Queiroz, and Z. Fan, "A robust technique for image descreening based on the wavelet transform," *IEEE Trans. Signal Processing*, vol. 46, pp. 1179–1194, Apr. 1998.
[6] S. Hein and A. Zakhor, "Halftone to continuous-tone conversion of error-diffusion coded images," *IEEE Trans. Image Processing*, vol. 4, pp. 208–216, Feb. 1995.
[7] P. Wong, "Inverse halftoning and kernel estimation for error diffusion," *IEEE Trans. Image Processing*, vol. 4, pp. 486–498, Apr. 1995.
[8] R. Stevenson, "Inverse halftoning via MAP estimation," *IEEE Trans. Image Processing*, vol. 6, pp. 574–583, Apr. 1997.
[9] Z. Xiong, M. Orchard, and K. Ramchandran, "Inverse halftoning using wavelets," in *IEEE Trans. Image Processing*, Oct. 1999, pp. 1479–1483.
[10] P. Roetling, "Image processing system and method employing adaptive filtering to provide improved reconstruction of continuous tone images from halftone images including those without screen structure," U.S. Patent 5 343 309, Aug. 1994.
[11] S. Mallat and S. Zhong, "Characterization of signals from multiscale edges," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 710–732, July 1992.
[12] T. D. Kite, N. Damera-Venkata, B. L. Evans, and A. C. Bovik, "A high quality, fast inverse halftoning algorithm for error diffused halftones," in *Proc. IEEE Conf. Image Processing*, Oct. 1998, pp. 59–63.
[13] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 12, pp. 629–639, July 1990.
[14] Z. Fan and R. Eschbach, "Limit cycle behavior of error diffusion," *Proc. IEEE Conf. Image Processing*, vol. 2, pp. 1041–1045, Nov. 1994.
[15] T. Huang, J. Burnett, and A. Deczky, "The importance of phase in image processing filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 23, pp. 529–542, Dec. 1975.
[16] P. Gill, W. Murray, and M. Wright, *Practical Optimization*. New York: Academic, 1981.
[17] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll, "Image selective smoothing and edge detection by nonlinear diffusion," in *SIAM J. Numer. Anal.*, Feb. 1992, vol. 29, pp. 182–193.
[18] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 8, pp. 679–698, Nov. 1986.
[19] T. D. Kite, B. L. Evans, and A. C. Bovik, "Modeling and quality assessment of halftoning by error diffusion," *IEEE Trans. Image Processing*, vol. 9, pp. 909–922, May 2000.

Thomas D. Kite received the B.S. degree in physics from Oxford University, Oxford, U.K., in 1991, and the M.S. and Ph.D. degrees in electrical engineering from The University of Texas, Austin, in 1993 and 1998, respectively. His M.S. thesis was in digital audio and his Ph.D. thesis was in image halftoning.

He is now a DSP Engineer with Audio Precision, Beaverton, OR.

Niranjan Damera-Venkata (SM'99) received the B.S.E.E. degree from the University of Madras, Madras, India, in July 1997 and the M.S.E.E. degree from The University of Texas, Austin, in May 1999. He is currently pursuing the Ph.D. degree in electrical engineering at The University of Texas.

He is currently with Hewlett-Packard Research Laboratories, Palo Alto, CA. His research interests include document image processing, symbolic design and analysis tools, image and video quality assessment, and fast algorithms for image processing.

Mr. Damera-Venkata won a 1998–1999 Texas Telecommunications Engineering Consortium Graduate Fellowship from The University of Texas. He is a member of Sigma Xi.

Brian L. Evans (S'88–M'93–SM'97) received the B.S.E.E.C.S. degree from the Rose-Hulman Institute of Technology, Terre Haute, IN, in May 1987, and the M.S.E.E. and Ph.D. degrees from the Georgia Institute of Technology, Atlanta, in December 1988 and September 1993, respectively.

From 1993 to 1996, he was a Postdoctoral Researcher at the University of California, Berkeley, where he worked on electronic design automation for embedded systems as a member of the Ptolemy project. He is the primary architect of the *Signals and Systems Pack* for Mathematica, which has been on the market since October 1995. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, The University of Texas, Austin. He is the Director of the Embedded Signal Processing Laboratory within the Center for Vision and Image Sciences. His research interests include real-time embedded systems; signal, image and video processing systems; system-level design; symbolic computation; and filter design. He developed and currently teaches multidimensional digital signal processing, embedded software systems, and real-time digital signal processing laboratory.

Dr. Evans is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, a member of the Design and Implementation of Signal Processing Systems Technical Committee of the IEEE Signal Processing Society, and the recipient of a 1997 National Science Foundation CAREER Award.

Alan C. Bovik (S'80–M'80–SM'89–F'96) received the B.S., M.S., and Ph.D. degrees in electrical engineering in 1980, 1982, and 1984, respectively, from the University of Illinois, Urbana-Champaign.

He is currently the General Dynamics Endowed Fellow and Professor with the Department of Electrical and Computer Engineering and the Department of Computer Sciences, The University of Texas, Austin (UT Austin). At UT Austin, he is also the Associate Director of the Center for Vision and Image Sciences, which is an independent research unit that brings together electrical engineering, computer science, and psychology professors, staff, and students. His current research interests include digital video, image processing, computer vision, wavelets, three-dimensional microscopy, and computational aspects of biological visual perception. He has published over 250 technical articles in these areas and holds U.S. patents for the image and video compression algorithms VPIC and VPISC.

Dr. Bovik is the Editor-in-Chief of the IEEE TRANSACTIONS ON IMAGE PROCESSING and is on the Editorial Board for PROCEEDINGS OF THE IEEE. He is the Founding General Chairman, First IEEE International Conference on Image Processing, which was held in Austin in November 1994.